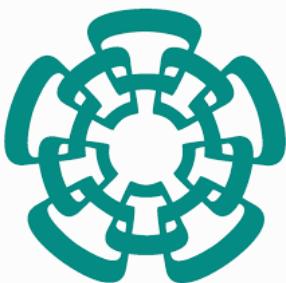


Centro de Investigación y Estudios Avanzados del I.P.N.



Reconocimiento de 8 herramientas

Materia: Análisis de Imágenes Digitales

Catedrático: Dr. Wilfrido Gómez-Flores

Alumno: Luis Alberto Ballado Aradias

28 de agosto de 2023

Índice

1	Introducción	3
2	Adquisición de la imagen	4
3	Preprocesamiento	6
4	Segmentación	7
4.1	Operaciones morfológicas	9
4.2	Erosión y dilatación	9
4.3	Rellenado de orificios	10
5	Extracción de características	11
5.0.1	Geométricas básicas	11
5.0.2	Rasgos basados en esqueleto	12
5.0.3	Momentos invariantes Hue	12
5.0.4	Cerco convexo	13
6	Clasificación	14
7	Resultados	15
8	Conclusiones	16
9	Apendice	18
9.1	Script de rasgos	18
9.2	Script de data augmentation	19
10	Experimentando en python	20

Lista de códigos

1	Factor de brillo	6
2	Script de rasgos	18
3	Python example	19
4	experimentación python	20
5	Python example	20

Lista de figuras

1	Etapas básicas para el análisis de imágenes.	4
2	Dimmer para el ajuste de la iluminación.	4
3	Cubo para la adquisición de imágenes.	5
4	Mejoramiento de contraste.	6
5	Umbralado con método de otsu.	7
6	Umbralado con K-Means.	8
7	Umbralado con método de entropía.	8
8	Detección de bordes con canny edges.	9
9	Imágenes segmentadas para obtención de rasgos geométricos básicos.	12
10	Rasgos basados en esqueleto.	12
11	Rasgos cerco convexo.	13
12	Umbralado con método de entropía.	13
13	Proceso de segmentado primera parte.	15
14	Proceso de segmentado última parte.	16
15	Imágenes diferentes fondos.	17
16	Proceso de segmentado última parte.	18

1. Introducción

El Análisis de Imágenes Digitales es un área fascinante que engloba muchas áreas de las ciencias computacionales como Optimización y Aprendizaje de máquinas.

El procesamiento de imágenes es un área multidisciplinaria que tiene como objetivo la manipulación y análisis de la información de una imagen en su parte más reducida conocida como pixel (picture element). Emergiendo una gran cantidad de técnicas y metodologías para el procesamiento y manipulación de imágenes.

Con el aumento de imágenes digitales, el procesamiento se ha vuelto necesario para aplicaciones como en robótica, que la segmentación de imágenes puede facilitar la navegación para un robot aéreo [5], en el diagnóstico de cáncer ayuda a los médicos en dar diagnósticos más acertados [3]. Asimismo, en la parte del entrenamiento con implementación de realidad aumentada [1] .

La detección y reconocimiento que son parte del área de visión por computadora, se han vuelto cada vez más utilizadas en aplicaciones como traductores de idiomas, análisis médicos, reconocimiento de rostros faciales.

La detección y categorización de objetos con imágenes se descompone de los siguientes procesos:

1. **Adquisición de la imagen.-** La toma de fotografía se realiza mediante una cámara que cuente con un sensor que puede ser de tipo CMOS, el sensor convierte la captura de la escena en señales digitales que son guardadas como una imagen.
2. **Preprocesamiento.-** Mediante las técnicas existentes se busca el mejoramiento del contraste, realzar características reduciendo el ruido, para hacer de la imagen lo más limpia posible para el paso de segmentación.
3. **Segmentación.-** Es la obtención de las regiones de interés que ayuda a la extracción de características. Ésta imagen binaria debe tomar la silueta de la figura lo más aproximada posible.
4. **Extracción de características.-** Éste proceso busca describir en cierta forma la imagen, describiendo valores de forma geométrica, colores, texturas y valores invariantes que se mantienen a pesar de rotaciones o escalamientos.
5. **Clasificación.-** Asigna una clase a cada región de interés dentro de la imagen basados en los rasgos extraídos en el paso anterior.

El proyecto busca el reconocimiento de ocho herramientas, aplicando las técnicas vistas en clase con el uso de MATLAB.

La figura 1 muestra las etapas a desarrollar para el reconocimiento de las ocho herramientas.

Una imagen $f(x, y)$ se puede modelar como el producto de dos componentes $f(x, y) = i(x, y), r(x, y)$, donde $0 < i(x, y) < \infty$ es la componente de iluminación y $0 < r(x, y) < 1$ es la componente de reflectancia.

Las dificultades que se tienen dentro del procesamiento de imágenes digitales son la iluminación del ambiente, siendo los sensores perceptibles a la iluminancia al obtener la imagen.

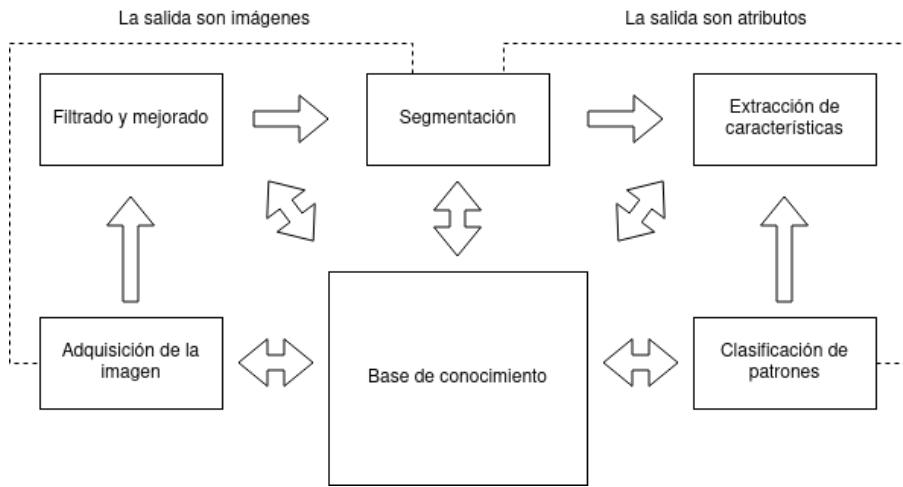


Figura 1: Etapas básicas para el análisis de imágenes.

2. Adquisición de la imagen

La adquisición de las imágenes se realizó con las siguientes herramientas:

- Cámara web Logitech C920s HD Pro 1080p webcam
- Cubo para la toma de fotografías marca Puluz, ver figura 3
- Control de iluminación dimmer, ver figura 2



Figura 2: Dimmer para el ajuste de la iluminación.

La cámara web cuenta con un sensor de tipo CMOS de 2-mega pixel [2], obteniendo imágenes limpias y muy bien contrastadas.

El objetivo del proyecto, es el reconocer automáticamente ocho herramientas que son:

- Martillo
- Desarmador
- Cinta de medir
- Llave perica
- Tijeras
- Pinza de punta
- Pinza eléctricas
- Pinza de presión

La toma de fotografías se realiza en un ambiente controlado como se muestra en la figura 3, ajustando la iluminación con un dimmer mostrado en la figura 2.



Figura 3: Cubo para la adquisición de imágenes.

Se usaron transformaciones geométricas para modificar la relación espacial entre píxeles en una imagen, usando propiedades que no deformen al objeto de interés como:

- Rotaciones
- Traslaciones

Logrando crear imágenes artificiales, de manera que sirvan para el aumento del conjunto de imágenes.

3. Preprocesamiento

Al contar con imágenes de buena calidad, solo se mejoró el contraste para reducir las sombras que las herramientas pueden formar y lograr obtener el contorno que representa más la imagen a reconocer.

El mejoramiento es la manipulación de una imagen, de tal forma que el resultado sea más útil que la imagen original para una aplicación en particular.

Una transformación de intensidad nos ayuda a modificar el contraste. Dentro del proyecto se aplicó una transformación de intensidad con un escalar de 1.5, para algunas imágenes más oscuras el valor de 2 resulta mejor para la detección de bordes más cercanos al objeto.



(a) Imagen original



(b) Imagen modificada del brillo

Figura 4: Mejoramiento de contraste.

De esta forma pudimos darle un alto contraste a la imagen como se puede apreciar en la figura 4.

A partir de la imagen original pasada a double¹, esta imagen se multiplica por un escalar de brillo para tener la imagen con el brillo alto.

```
1 %% pasar a double
2 original_gris = double(I);
3 %% multiplicar por el factor de ruido
4 img_ajustada = original_gris * factor_brillo;
```

Código 1: Factor de brillo

A pesar de que los objetos en su mayoría poseen partes cromadas, se logran contrastar con el fondo original blanco. Logrando apreciar mejor los objetos, este paso es de gran ayuda. Siendo el factor de brillo un atributo dentro de la función de segmentación.

¹se hace un cambio de estructura a double

4. Segmentación

Se hacen pruebas con diversas técnicas de segmentación, optando por un resultado sencillo con uso de operaciones morfológicas en base a una imagen de detección de bordes con Canny Edges.

De igual forma se exploraron otras técnicas que se listan a continuación.

- Método de umbralado global **Método de Otsu**

Este método sería el indicado, pero las partes color cromo en los objetos afecta en el umbralado de la imagen, haciendo perder rasgos representativos de los objetos como las puntas en la tijera, las partes metálicas del desarmador o de las pinzas. La figura 5 ilustra el resultado después de aplicar éste método (Otsu).



(a) Imagen original escala de grises



(b) Imagen umbralada con el método de otsu

Figura 5: Umbralado con método de otsu.

- Método de segmentación con **K-Means** (K-Means clustering)

El algoritmo k-means es una técnica de agrupamiento que divide un conjunto de datos en k grupos o clusters. Los datos se agrupan de modo que los puntos de un grupo sean más similares que los puntos de otros grupos.

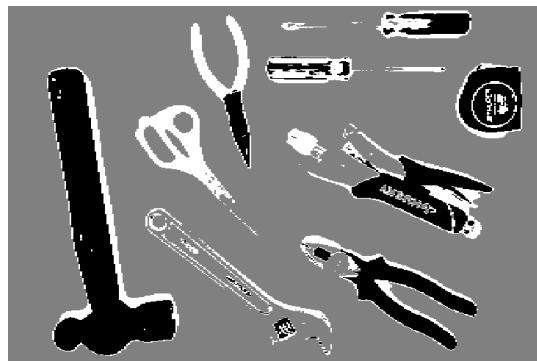
A partir de una simple idea de crear dos grupos, el fondo y los objetos. Éste método sirve igual de bien que el Método de Otsu, pero en nuestro caso, el problema con el color cromo de los objetos hace que los valores cromo formen parte del conjunto del fondo. La figura 6 ilustra el resultado después de aplicar éste método (K-Means clustering).

- Visualización de la **Entropía**

Con una segmentación mediante el método de entropía se puede obtener la información de la presencia de los objetos, lamentablemente al estar con mucha información no fué utilizada en el proyecto. Pero replanteandolo nuevamente, se puede hacer algún filtro para quedarse solo con los píxeles blancos con algún método como lógica difusa. La figura 12 ilustra el resultado después de aplicar éste método (Entropía).



(a) K-Means con dos clases



(b) K-Means con tres clases

Figura 6: Umbralado con K-Means.



(a) Imagen original escala de grises



(b) Imagen después del método de entropía

Figura 7: Umbralado con método de entropía.

- Detección de bordes **Canny Edges**

Al contar con una representación limpia de los objetos de interés, la detección de los objetos a partir de sus bordes es una buena opción para lidar con las piezas cromadas. La figura 8 ilustra el resultado después de aplicar éste método.

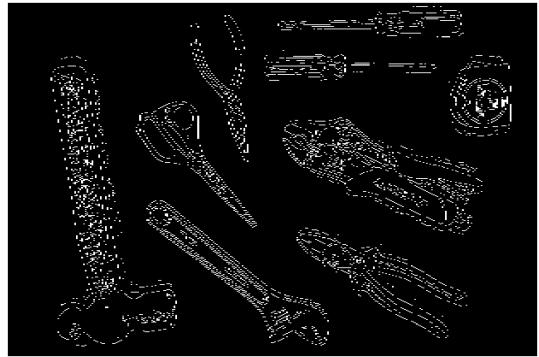
Probablemente sea el algoritmo para detección de bordes más usado para reconocer bordes en visión por computadora.

Usa las mejores propiedades del operador del gradiente y el operador laplaciano. Básicamente los pasos que sigue son los siguientes:

- Suaviza la imagen con un filtro gausiano 2D
- Calcula el gradiente de la imagen usando un operador de sobel de 3x3, dando resultados de las derivadas para cada punto para el eje x y el eje y
- A partir de esos valores se puede calcular la magnitud para cada píxel, así como su orientación
- Se calcula el Laplaciano a lo largo de la dirección del gradiente
- Se usan los cruces por cero en el Laplaciano para encontrar la ubicación del borde



(a) Imagen original escala de grises



(b) Detección de bordes con canny edges

Figura 8: Detección de bordes con canny edges.

En matlab, el filtro de **Canny Edges** detecta bordes buscando los máximos locales del gradiente de la imagen. Las funciones de borde utilizan la derivada de un filtro gaussiano para calcular el gradiente. Este método utiliza dos umbrales para detectar bordes fuertes y débiles e incluye bordes débiles en la salida si están asociados con bordes fuertes. Debido a que el método de Canny utiliza dos umbrales diferentes, es menos propenso a errores debido al ruido que otros métodos y es más probable que detecte bordes realmente débiles [4].

- Sigma valor escalar que especifica la desviación estándar del filtro gaussiano. El valor predeterminado es $\text{sqrt}(2)$. automáticamente en función de sigma [4].
- Umbral de sensibilidad es especificado como escalar numérico, la función en matlab ignora todos los bordes cuya intensidad no es mayor que threshold indicado, al no indicar un valor la función escoge los valores automáticamente [4].

Al considerar la última técnica con la obtención de los bordes, debemos cerrar la imagen con elementos estructurantes, donde podemos encontrar la línea, ésta puede tomar valores de inclinación a ciertos grados y ser de 4 ó 8 vecindad.

4.1. Operaciones morfológicas

La erosión y dilatación, operadores que fueron aplicados en el proyecto.

- Apertura: una erosión seguida de una dilatación $f \circ b = (f \ominus b) \oplus b$
- Cerradura: una dilatación, seguida de una erosión $f \bullet b = (f \oplus b) \ominus b$

El borde de un conjunto A que contiene los píxeles del primer plano se obtiene como:

$$B(A) = A - (A \ominus b)$$

donde A es la figura binaria y b el elemento estructurante.

4.2. Erosión y dilatación

Explicar las partes de erosion y dilatacion

4.3. Rellenado de orificios

Un orificio es una región del fondo rodeado por un borde conexo de pixeles del primer plano.

Se efectua una reconstrucción por dilatación.

Contorno de una región

El contorno de una región conexa es el conjunto de pixeles que tienen al menos en pixel vecino que corresponde al fondo en 4 ó 8 adyacencia.

Buscar que tipo de etiquetado usa BWLABEL .- QUE ALGORITMO?

Etiquetado de regiones conexas

5. Extracción de características

El reconocimiento automático de objetos requiere calcular rasgos que describan propiedades físicas de los objetos.

Un rasgo (atributo o característica), es un valor numérico que cuantifica alguna propiedad de forma, textura, color, geometría, etc.

- Un rasgo debe ser discriminante, invariante, incorrelado y rápido de computar.
- Los rasgos de forma generalmente se calculan a partir de la segmentación del objeto y se divide entre todos los pixeles de las regiones de interés.
- Basados en región distribución de todos los pixeles.
- Basados en contorno, variaciones a lo largo del contorno.

La extracción de características que se utilizaron en el proyecto son las siguientes:

1. Rasgos geométricos básicos
2. Momentos de Hue
3. Cerco convexo
4. Esqueletización

5.0.1. Geométricas básicas

El área es el número total de pixeles que cubre la región del objeto.

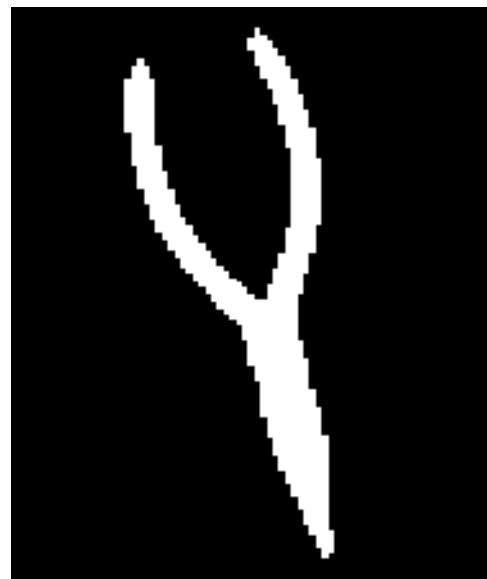
El perímetro es la longitud del contorno del objeto. El resultado depende del tipo de conectividad que pueda tener. (4 ó 8 vecindad).

Para el cálculo de rasgos geométricos básicos, se hace uso del área y perímetro de la figura, el cual se obtiene mediante la imagen segmentada iterando entre todas las etiquetas.

- | | |
|----------------|-------------------|
| ■ Redondez | ■ Compacidad |
| ■ Circularidad | ■ Factor de forma |



(a) Tijera



(b) Pinza de punta

Figura 9: Imágenes segmentadas para obtención de rasgos geométricos básicos.

5.0.2. Rasgos basados en esqueleto

Otra generación de rasgos usada en el proyecto, son los rasgos por esqueletizar la imagen.



(a) Imagen umbralizada



(b) Imagen esqueletizada

Figura 10: Rasgos basados en esqueleto.

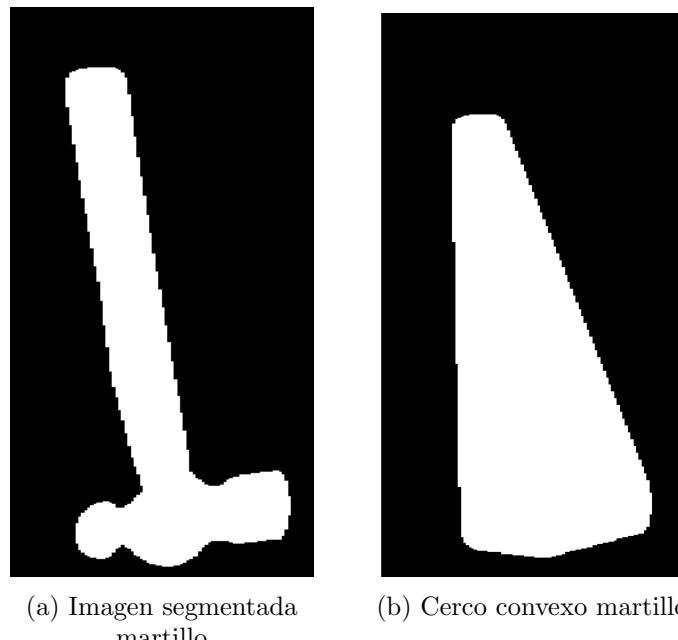
- Puntos finales
- Ramas
- Número de pixles
- Área
- Perímetro

5.0.3. Momentos invariantes Hue

Los momentos son proyecciones de una función sobre una base polinomial usados para medir la distribución de masa de un cuerpo.

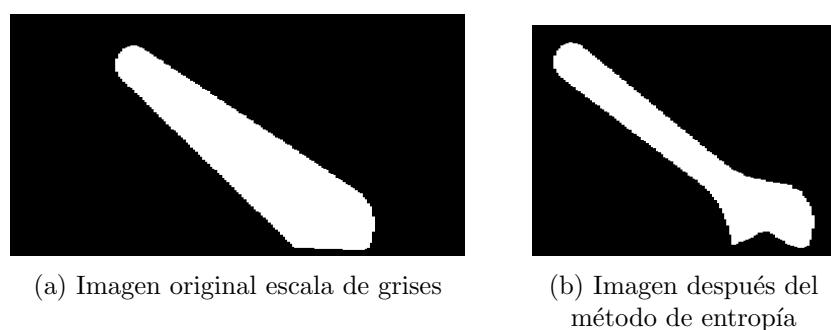
5.0.4. Cercos convexos

Los momentos son proyecciones de una función sobre una base polinomial usados para medir la distribución de masa de un cuerpo.



(a) Imagen segmentada martillo
(b) Cercos convexos martillo

Figura 11: Rasgos cerco convexo.



(a) Imagen original escala de grises
(b) Imagen después del método de entropía

Figura 12: Umbralado con método de entropía.

- Solidez
- Convexidad

6. Clasificación

Recordando las etapas del reconocimiento de objetos en imágenes.

1. Sensado.- Capturar la imagen.
2. Preprocesamiento.- Mejorar la calidad de la imagen y segmentar los objetos de interés.
3. Extracción de características.- Describir los objetos con rasgos cuantitativos discriminantes e invariantes.
4. Clasificación.- Asignar una etiqueta de clase a cada objeto.

Obtener un vector de patrones.

Para el proyecto se consideran ocho clases siendo representadas por los objetos siguientes:

- | | |
|------------------|--------------------|
| ■ Martillo | ■ Tijeras |
| ■ Desarmador | ■ Pinza de punta |
| ■ Cinta de medir | ■ Pinza eléctricas |
| ■ Llave perica | ■ Pinza de presión |

En este proyecto se usa el método de datos de entrenamiento que son vectores de patrones asociados a una etiqueta de clase (x,y) con $y \in \Omega = w_1, \dots, w_c$

La selección del algoritmo de clasificación debe tener las siguientes cualidades.

- Generación de fronteras de decisión no lineales.
- Clasificación en más de dos clases (multiclas).
- Entrenamiento en un tiempo de cómputo razonable.

Se elige el clasificador k-nn de vecinos más cercanos por su simpleza.

De modo que el reconocimiento de un objeto se realiza mediante la comparación entre el objeto de entrada y las muestras que se hayan obtenido en la etapa de extracción de características.

Al pasar por su segmentación y analizar los rasgos obtenidos, se asigna al grupo de mayor parecido. Entonces, el objeto cercano indica que son parecidos y la similitud se reducirá a medida que no tenga una similitud.

Diferentes distancias se pueden implementar en el algoritmo de los k-vecinos más cercanos, siendo la distancia euclíadiana la que se utilizó en el desarrollo del proyecto. El parámetro k de vecinos, es el número de muestras de entrenamiento cercanas con las que se comparan.

El algoritmo KNN realiza los siguientes pasos:

1. Medir la distancia Euclíadiana entre el objeto de interés y las muestras obtenidas la extracción de rasgos.
2. Identificar los patrones k más cercanos y obtener la etiqueta de la clase a la que pertenece.
3. Asignación de la clase al objeto de interés.

7. Resultados

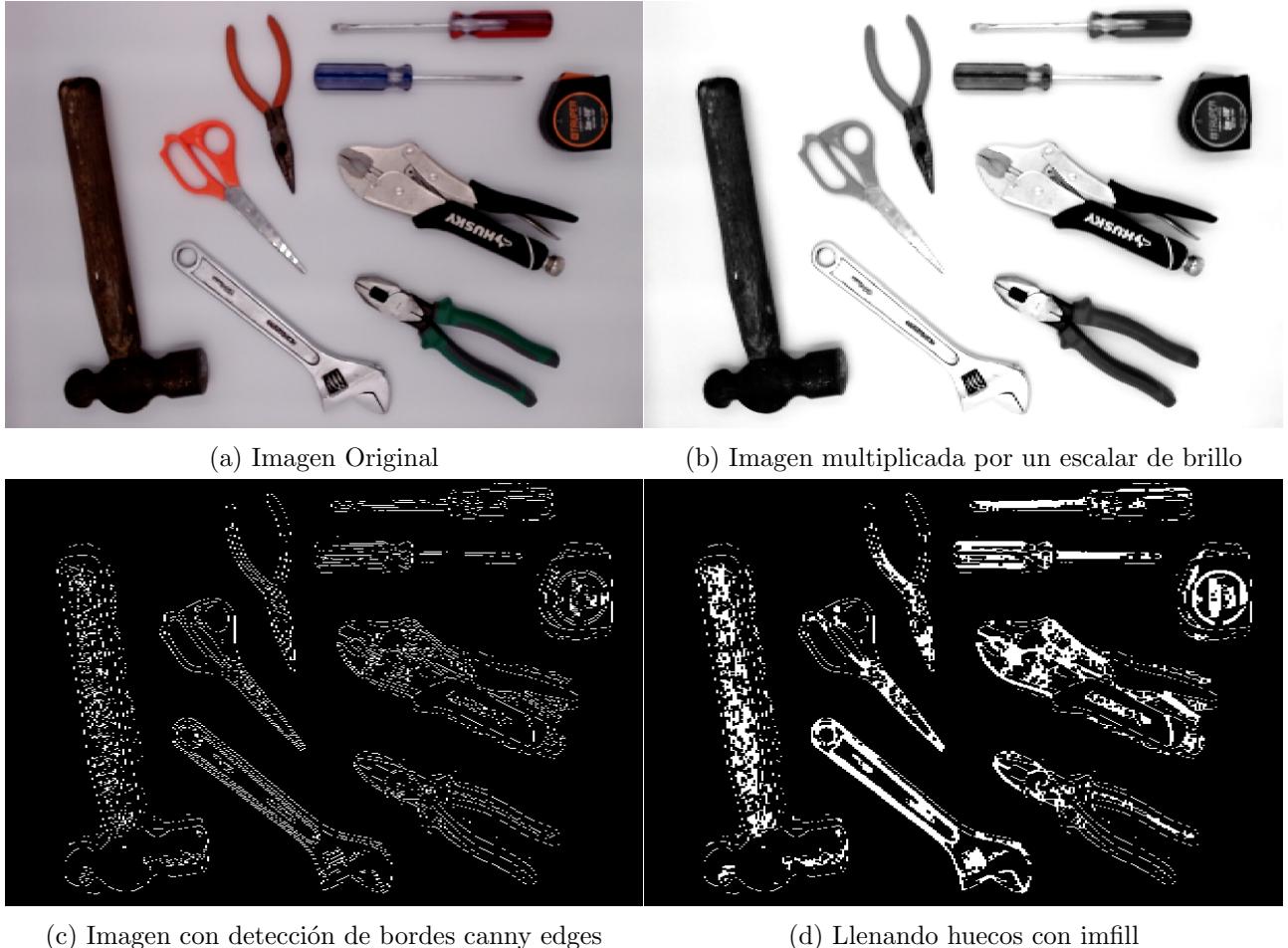


Figura 13: Proceso de segmentado primera parte.

Dentro de las imágenes, la similitud de esqueletos entre la cinta, la llave perica, tijera, pinzas de punta y electrica que tienen formas similares.

La mayoría de desarmadores no tiene ramas y tiene dos puntos finales, lo que lo confunde con una cinta al solo considerar las ramas y los puntos terminales se tiene una exactitud del 20 %, mientras al agregar el num de pixeles crece al 78 %, agregando más información como el área y perímetro la probabilidad de acierto crece aún más, presentándose valores arriba del 90 %.

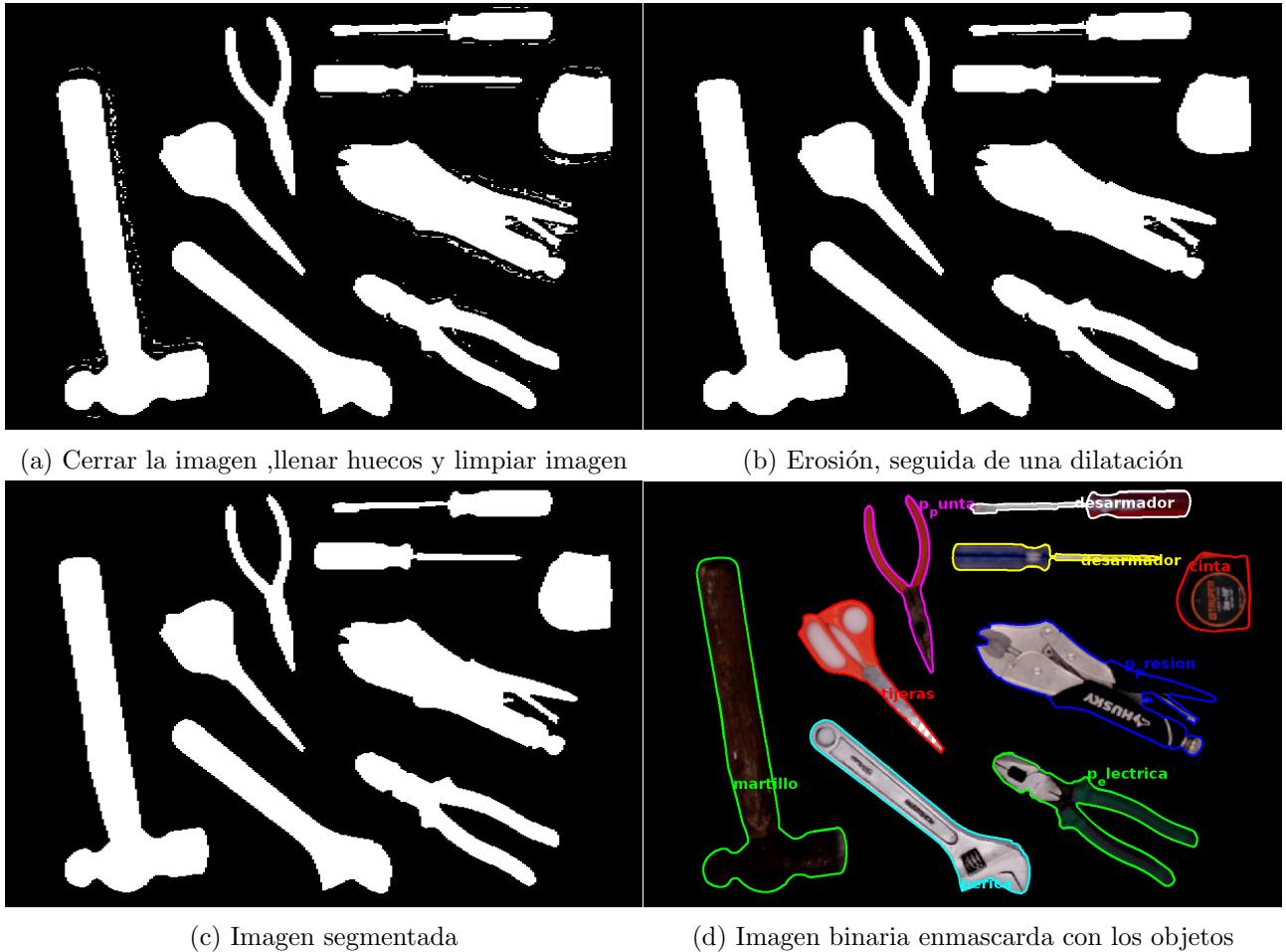


Figura 14: Proceso de segmentado última parte.

8. Conclusiones

Estudios tempranos de Inteligencia Artificial buscaban duplicar los pensamientos humanos, pero ahora los estudios recientes muestran la tendencia en replicar el resultado y las computadoras actúan como sistemas expertos.

Las computadoras son dispositivos simbólicos capaces de manipular cualquier tipo de símbolos siendo los números una clase importante, pero las computadoras son más generales que eso. Sabemos de la generalidad de la computación desde los tiempos de Alan Turing en los 1930's y se tienen intuiciones que Babbage tuvo también estudios que fueron afirmados por Ada Lovelace, en 1842 Ada Lovelace escribió sobre la ingeniería analítica de Babbage que buscaba unir el mundo mecánico con el mundo de las cosas abstractas, en la psicología moderna es llamado The physical symbol system hypothesis y son las bases en que la Inteligencia Artificial funciona.

La Inteligencia Artificial como ciencia emplea el uso de computadoras para procesar conocimientos simbólicos usando métodos de inferencias lógicas, en otras palabras, nos referimos a inferencia y no cálculos que se piensa en el sentido tradicional, hablamos de conocimiento y no números como en la forma tradicional.

Al poner conocimiento en programas computacionales que para los humanos nos es fácil o a veces un reto intelectual y el conocimiento que le pasamos es representativo en cierta forma particular.



(a) Imagen objetos fondo negro



(b) Resultado fondo negro, hue 100 %



(c) Imagen objetos con fondo rojo



(d) Resultado fondo rojo, cerco convexo 89 %

Figura 15: Imágenes diferentes fondos.

Por otra parte, los sistemas expertos basados en conocimientos son aplicables en cualquier área en que sea especializado el conocimiento y sea rutinario la toma de decisiones, estrategias para resolver problemas, diagnosticos, .. etc.

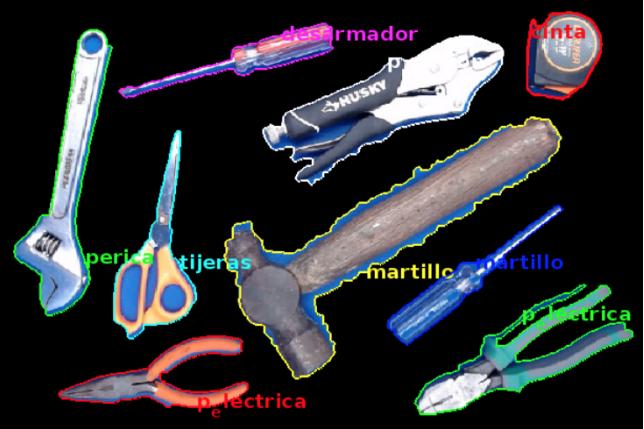
Siendo de gran ayuda en un gran rango de aplicaciones, el **conocimiento medico**, en clasificación de un experto, puede estar reflejado con el funcionamiento de una red neuronal.

Para algun diagnostico médico ó segmentadores más fáciles, pero igual de complicados como la segmentación en imágenes para la creación de sistemas autonomos.

Programas que se puedan ejecutar en cualquier computadora o dispositivo que permita la ejecución de código multiplataforma como python ó C++, que tienen capacidades altas de manipulación simbolica, la toma de decisiones es primordial en una inteligencia basada en conocimiento y esa manipulación simbolica es necesaria.



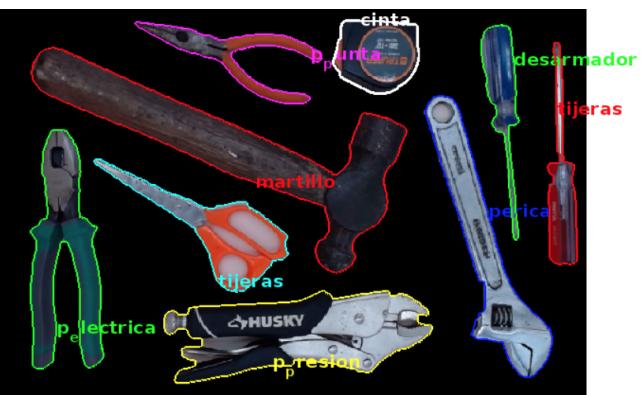
(a) Imagen objetos fondo azul



(b) Resultado fondo azul, rasgos geometricos 97 %



(c) Imagen objetos fondo claro



(d) Resultado fondo claro, cerco convexo 89 %

Figura 16: Proceso de segmentado última parte.

9. Apendice

9.1. Script de rasgos

La obtención de rasgos se realizó mediante un script que itera entre todas las carpetas de las imágenes, haciendo rotaciones y traslaciones aleatorias, buscando mantener la imagen, por lo que se le añade un efecto para que el fondo se replique

```

1  %% Sample Matlab code
2 !mv test.txt test2.txt
3 A = [1, 2, 3; ... foo
4     4, 5, 6];
5 s = 'abcd';
6 for k = 1:4
7     Disp(s(k)) % bar
8 end
9 %
10 create row vector x, then reverse it
11 %
12 x = linspace(0,1,101);

```

```
13 y = x(end:-1:1);
```

Código 2: Script de rasgos

9.2. Script de data augmentation

We did some experiments ...

```
1 %% Sample Matlab code
2 !mv test.txt test2.txt
3 A = [1, 2, 3; ... foo
4     4, 5, 6];
5 s = 'abcd';
6 for k = 1:4
7     Disp(s(k)) % bar
8 end
9 %
10 create row vector x, then reverse it
11 %
12 x = linspace(0,1,101);
13 y = x(end:-1:1);
```

Código 3: Python example

10. Experimentando en python

```
1 import numpy as np
2
3 def incmatrix(genl1,genl2):
4     m = len(genl1)
5     n = len(genl2)
6     M = None #to become the incidence matrix
7     VT = np.zeros((n*m,1), int) #dummy variable
8
9     #compute the bitwise xor matrix
10    M1 = bitxormatrix(genl1)
11    M2 = np.triu(bitxormatrix(genl2),1)
12
13    for i in range(m-1):
14        for j in range(i+1, m):
15            [r,c] = np.where(M2 == M1[i,j])
16            for k in range(len(r)):
17                VT[(i)*n + r[k]] = 1;
18                VT[(i)*n + c[k]] = 1;
19                VT[(j)*n + r[k]] = 1;
20                VT[(j)*n + c[k]] = 1;
21
22            if M is None:
23                M = np.copy(VT)
24            else:
25                M = np.concatenate((M, VT), 1)
26
27    VT = np.zeros((n*m,1), int)
28
29 return M
```

Código 4: experimentación python

```
1 import numpy as np
2
3 def incmatrix(genl1,genl2):
4     m = len(genl1)
5     n = len(genl2)
6     M = None #to become the incidence matrix
7     VT = np.zeros((n*m,1), int) #dummy variable
8
9     #compute the bitwise xor matrix
10    M1 = bitxormatrix(genl1)
11    M2 = np.triu(bitxormatrix(genl2),1)
12
13    for i in range(m-1):
14        for j in range(i+1, m):
15            [r,c] = np.where(M2 == M1[i,j])
16            for k in range(len(r)):
17                VT[(i)*n + r[k]] = 1;
18                VT[(i)*n + c[k]] = 1;
19                VT[(j)*n + r[k]] = 1;
20                VT[(j)*n + c[k]] = 1;
21
22            if M is None:
23                M = np.copy(VT)
24            else:
25                M = np.concatenate((M, VT), 1)
26
27    VT = np.zeros((n*m,1), int)
```

```
28  
29     return M
```

Código 5: Python example

Referencias

- [1] R. Aggarwal and A. Singhal. Augmented reality and its effect on our life. In *2019 9th International Conference on Cloud Computing, Data Science and Engineering (Confluence)*, pages 510–515, 2019.
- [2] CDW. logitech c920s HD. <https://www.cdw.com/product/logitech-hd-pro-webcam-c920s-webcam/5479466>, 2023. [Online; accessed 27-August-2023].
- [3] M. N. R. Devi, A. Kumar, G. Swetha, U. S. Chavan, and V. M. Davasam. Cancer detection using image processing and machine learning. In *2022 International Conference on Artificial Intelligence and Data Engineering (AIDE)*, pages 96–100, 2022.
- [4] MatLab. edge. <https://la.mathworks.com/help/images/ref/edge.html#buo5g3w-1-threshold>, 2023. [Online; accessed 27-August-2023].
- [5] N. Smolyanskiy, A. Kamenev, J. Smith, and S. Birchfield. Toward low-flying autonomous mav trail navigation using deep neural networks for environmental awareness, 2017.